SenseLearner: Minimally Supervised Word Sense Disambiguation for All Words in Open Text (2005)

Rada Mihalcea and Ehsanul Faruque

Presented by
Michael Gabilondo

Main Idea

- SenseLearner: WSD for all verbs, adjectives and nouns
- Initially trains on sense annotated corpus to learn from local context around each verb, adjective and noun
- **(1)** For words in unannotated corpus that **have** been seen
  - Learns a model for each POS from annotated corpus
- **(2)** For words in unannotated corpus that **have not** been seen
  - Uses syntactic dependencies as features of local context, e.g. verb-obj
  - Generalizes concepts learned by using WordNet to extend coverage to all words (100% coverage)
- **(1) = Semantic Language Model** and **(2) = Semantic Generalizations** are done sequentially.

SemCor

- Small Manually Annotated Corpus: Over 200,000 content words
- All content words tagged with sense corresponding to WordNet senses and POS
- Original uses WordNet 1.6 senses, but Rada Mihalcea has a version that is mapped to WordNet 2.0 senses on her website. Automatically mapped.
- Initial training data

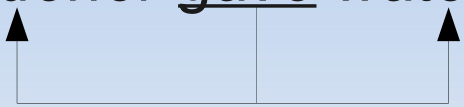
Semantic Language Model

- For each verb, adjective and noun in SemCor, build a feature vector with label **word#sense**
- Feature vector is added to noun training set, verb training set or adjective training set if it is a noun, verb or adjective, respectively
- Feature vectors in noun training set record the first noun, verb or adjective before the target noun, within a window of at most five words to the left, and its part of speech
  - *The gardener gave water to the plants.*

    Therefore, plant#2 would contain *water (noun)*. Each occurrence of plant#2 in SemCor would add to this feature vector.
  - Added 86,973 feature vectors to the noun training set this way

Semantic Language Model

- Feature vectors in verb model contain the first word before and the first word after the target verb, and its part of speech.
  - *The gardener gave water to the plants.*



  - Therefore, give#24 contains (gardener-Noun, water-Noun).
  - 47,838 vectors constructed from annotated corpus

Semantic Language Model

- There are two adjective models that are applied sequentially and then ”combined through voting”:

- One relying on the first noun after the target adjective, within a window of at most five words.
- A second model relying on the first word before and the first word after the target adjective, and its part of speech.

- 35,335 vectors in each of the two adjective models were created from SemCor.

Annotating new text with Semantic Language Model

- After the vectors created from the annotated corpus are added to the noun, verb and adjective models, similar vectors are created from the unannotated test corpus: each is labeled **only with a word** and uses the features described before.
- A separate learning process using Timbl is done for each part of speech (Timburg Memory based learner). Presumably, it learns how to label feature vectors with a word and sense.
- Then, the learning algorithm labels each vector in the test data set with a *predicted* word and sense.
- If the predicted word matches the word in the word the test feature vector was labeled with, then the vector is labeled with the predicted sense.
- This covered 85.6% of words in SENSEVAL-3 English all-words data set. Precision for this model alone was not given.

Semantic Generalizations

- There is a second model that covers the words not labeled by the Semantic Language Model: Semantic Generalizations using Syntactic Dependencies and a Conceptual Network

- Syntactic Dependencies are used as Local Context features:
  - e.g. in ”produced evidence” there is a verb-object relationship between *produce#i* and *evidence#j*, where i and j are the senses. Therefore, this dependency supports senses i and j of produce and evidence, respectively, when they are encountered in a verb-object relationship.
  - Link Grammar Parser is used to find syntactic dependencies

```
Found 2 linkages (2 with no P.P. violations)
  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=10)

    +----------------------Xp----------------------+
    |                      +---------Os---------+   |
    +------Wd-------+      |     +-----Dmu-----+   |
    |      +---Ds--+---Ss---+     |     +---AN--+   |
    |      |       |       |     |     |       |   |
LEFT-WALL the gardener.n gave.v the plant.n water.n .

Constituent tree:

(S (NP The gardener)
   (VP gave
       (NP the plant water))
   .)
```

```
  Linkage 2, cost vector = (UNUSED=0 DIS=1 AND=0 LEN=10)

    +----------------------Xp----------------------+
    |                      +---------Osn--------+   |
    +------Wd-------+      +----Os----+         |   |
    |      +---Ds--+---Ss---+    +--Ds-+         |   |
    |      |       |       |     |     |       |   |
LEFT-WALL the gardener.n gave.v the plant.n water.n .

Constituent tree:

(S (NP The gardener)
   (VP gave
       (NP the plant)
       (NP water))
   .)
```

Semantic Generalizations

- Generalizations are done with WordNet: a conceptual network
- For ”She drank some water” Link Grammar Parser produces a verb-object relationship between *drank* and *water*.
- *take#18* is a hypernym of *drink#1*
- *liquid#1* is a hypernym of *water#6* and *tea#1* is a hyponym of *liquid#1*
- Therefore, SenseLearner also concludes that there is a verb-object relationship between *take#18* and *tea#1*.
- For ”She will take tea,” there is a verb-object relationship between *take* and *tea*. The above relationship supports sense 18 for *take* and sense 1 for *tea*.

Semantic Generalizations

- Training is done with SemCor
- 1. Remove SGML tags from SemCor to produce one sentence per line.
- 2. Parse the sentence with Link Parser and save all dependency pairs: subject-verb, determiner-noun, verb-object, etc.
- 3. Add back POS and sense information to words in pairs
- 4. Build a positive feature vector for each pair (word1, word2) that appears
  - (relationship, word1#POS#sense1, hypernyms of word1#sense1, word2#POS#sense2, hyperynms of word2#sense2)
- 5. Build negative feature vectors for depenency pairs for the remaining senses that did not appear

Semantic Generalizations

- Annotating the test corpus
- 1. Parse sentences using Link Parser and save dependency pairs
- 2. Start with the leftmost word in the sentence and get all the dependency pairs between it and any words it connects to
- 3. Create feature vectors for all possible combinations of senses. E.g. given pair (word1, word2), if word1 has 2 senses and word2 has 3 senses, create 6 feature vectors, one for each combination of senses
- 4. Pass all of these feature vectors to Timbl, which attaches either a positive or negative label to the vectors, based on information from the training data

Semantic Generalizations

- Example. Suppose the following sentence is in SemCor.
- *The Fulton County Grand Jury said Friday an investigation of Atlanta's recent primary election produced "no evidence" that any irregularities took place.*

  verb-obj
- Training phase starts getting all possible dependency pairs. Consider verb-obj pair (produce#v#4, evidence#n#1).
- Build a feature vector
  - (Os, produce#v#4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, produce#v#4, expose#v#3, show#v#4, evidence#n#1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, evidence#n#1, information#n#3, cognition#n#1, psychological feature#n#1)
- Consider sentence in Test Corpus: "expose meaningful information"
- Use above vector to tag expose#v#3 and information#n#3.

Evaluation

Class	Precision	Fraction of Recall
Nouns	69.4	31.0
Verbs	56.1	20.2
Adjectives	71.6	12.2
Total	64.6	64.6

Table 1: SENSELEARNER results in the SENSEVAL-3 English all words task

- 2081 content words
- Baseline of ”most frequent sense” was 60.9%
- Verbs were most difficult: many WordNet senses
- Precision = Recall, since there was 100% coverage

Conclusion

- **Pros**
  - Covers all adjectives, nouns, verbs.
  - Generalizes concepts using WordNet to cover unseen words
  - Attempts to handle all WordNet senses
  - Uses Link Parser to uncover many syntactic dependencies to use as local context for homographs: reveals subject-verb, verb-object, determiner-noun relationships, etc.
  - Training corpus mapped to WordNet senses
- **Cons**
  - Only 3.7% over baseline.
  - Individual precisions for Semantic Language Model and Semantic Generalizations not given
  - Anaylsis of errors or ideas for future improvement not given
  - Training corpus too small?? (Over 200,000 content words)